❖ *Introduction :*

-Lors de ce projet passionnant, j'ai décidé de revisiter et réinventer tout ce que nous avons appris en cours :

* Utiliser des tableaux, des listes, ...
* Utiliser les formulaires.
* JavaScript.
* Communiquer des données entre scripts HTML, utilisation de fonctions et gestion des évènements JavaScript.
* Utiliser les évènements JavaScript pour détecter les actions de l'utilisateur, et à manipuler les éléments HTML en utilisant les fonctions du DOM.
* JSON

❖ *Présentation du min projet :*

❖ *Problématique :*

De nos jours, on constate une tendance croissante des professeurs proposant des cours particuliers aux étudiants. Chaque professeur peut se retrouver avec une classe comprenant entre 200 et 300 étudiants. Cependant, il est essentiel pour le professeur de garder une trace des informations de chaque élève, telles que les paiements effectués pour les cours ou la fin de validité du mois payé, ainsi que leurs coordonnées pour faciliter les échanges. Faire cela manuellement avec un stylo et du papier, voire même avec Excel, peut s'avérer complexe pour gérer une telle quantité d'étudiants.

❖ *Objectifs :*

C'est là qu'intervient notre solution ! Nous avons développé un outil innovant qui simplifie et automatise la gestion de vos étudiants. Fini les tracas administratifs et les longues heures passées à effectuer des tâches fastidieuses. Grâce à notre plateforme intuitive, vous pouvez facilement enregistrer et suivre les informations de chaque étudiant en quelques clics.

Plus besoin de vous préoccuper des paiements en retard ou des coordonnées perdues. Tout est centralisé et organisé de manière claire et efficace.

Libérez-vous de la paperasse et maximisez votre temps pour ce qui compte vraiment : enseigner. Notre solution vous offre la tranquillité d'esprit et la productivité dont vous avez besoin pour fournir un enseignement de qualité à vos étudiants. Prêt à simplifier votre gestion de classe ? Essayez dès maintenant notre plateforme et transformez votre façon d'enseigner.

- ***Interface d'entrée :***

Plongez dans notre interface d'accueil captivante qui vous offre une expérience utilisateur exceptionnelle. Dès que l'administrateur se connecte, il est immédiatement dirigé vers une page dynamique offrant une vue complète de la liste des utilisateurs. Mais ce n'est pas tout ! L'administrateur a le pouvoir d'ajouter de nouveaux utilisateurs ou même de supprimer ceux existants, offrant ainsi un contrôle total sur la plateforme.

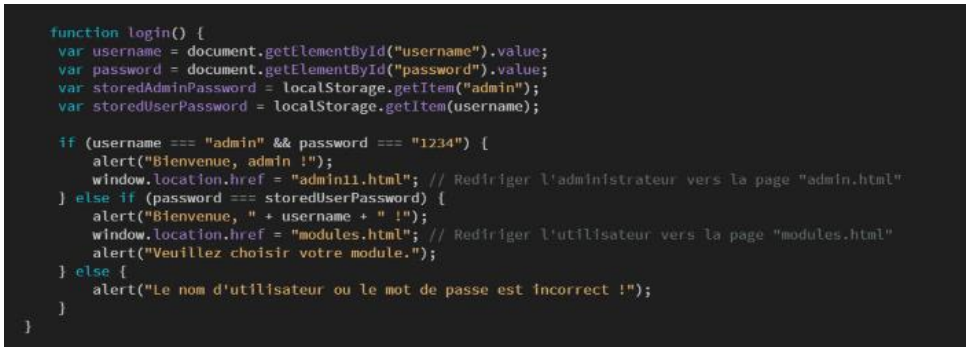
Une fois que l'administrateur a ajouté un nouvel utilisateur, celui-ci peut alors se connecter en toute simplicité en utilisant ses propres informations. Imaginez la fluidité et la praticité offertes par notre système, où chaque utilisateur peut accéder instantanément à son propre espace personnel.

Profitez d'une expérience sans encombrement administratif, avec un processus de gestion d'utilisateurs intuitif et efficace. Notre interface élégante vous permet de rester concentré sur ce qui compte vraiment : créer des connexions, faciliter l'apprentissage et stimuler la réussite de chaque individu.

En cas de besoin, nous avons prévu des options pratiques pour assurer une assistance immédiate. Si un utilisateur rencontre des difficultés, il peut simplement cliquer sur "Aide", ou bien sur "Contact" qui permet à l'utilisateur de communiquer directement avec l'administrateur.

En un seul clic, les utilisateurs pourront exprimer leurs préoccupations, poser des questions spécifiques ou demander une assistance personnalisée.

Voici le code JAVASCRIPT pour faire le login :



```
function login() {
    var username = document.getElementById("username").value;
    var password = document.getElementById("password").value;
    var storedAdminPassword = localStorage.getItem("admin");
    var storedUserPassword = localStorage.getItem(username);

    if (username === "admin" && password === "1234") {
        alert("Bienvenue, admin !");
        window.location.href = "admin11.html"; // Rediriger l'administrateur vers la page "admin.html"
    } else if (password === storedUserPassword) {
        alert("Bienvenue, " + username + " !");
        window.location.href = "modules.html"; // Rediriger l'utilisateur vers la page "modules.html"
        alert("Veuillez choisir votre module.");
    } else {
        alert("Le nom d'utilisateur ou le mot de passe est incorrect !");
    }
}
```

Le code fourni est une fonction JavaScript appelée "login()". Voici une explication de son fonctionnement :

1-La fonction "login()" est déclenchée lorsqu'un utilisateur clique sur le bouton de connexion.

2-Elle récupère les valeurs entrées par l'utilisateur pour le nom d'utilisateur et le mot de passe à l'aide des identifiants d'éléments HTML correspondants.

3-Ensuite, elle utilise la méthode "localStorage.getItem()" pour récupérer le mot de passe enregistré dans le stockage local (localStorage). Pour l'administrateur, le mot de passe est stocké avec la clé "admin". Pour les utilisateurs normaux, le mot de passe est stocké avec une clé correspondant à leur nom d'utilisateur.

4-La fonction vérifie d'abord si le nom d'utilisateur et le mot de passe correspondent à ceux de l'administrateur ("admin" et "1234" respectivement). Si c'est le cas, une alerte de bienvenue s'affiche pour l'administrateur et il est redirigé vers la page "admin.html" spécifiée.

5-Si le nom d'utilisateur et le mot de passe ne correspondent pas à ceux de l'administrateur, la fonction vérifie si le mot de passe

correspond à celui stocké pour l'utilisateur spécifié. Si c'est le cas, une alerte de bienvenue personnalisée s'affiche pour l'utilisateur, suivi d'une redirection vers la page "modules.html" spécifiée. Une autre alerte invite ensuite l'utilisateur à choisir son module.

6-Si ni les identifiants de l'administrateur ni ceux de l'utilisateur ne correspondent, une alerte est affichée indiquant que le nom d'utilisateur ou le mot de passe est incorrect.

En résumé, cette fonction de connexion vérifie les informations d'identification fournies par l'utilisateur, puis redirige l'administrateur ou l'utilisateur vers les pages appropriées en fonction des informations saisies.

❖ ***Page de l'administrateur :***

Cette page offre à l'administrateur une vue complète de la liste des utilisateurs. Mais ce n'est pas tout ! L'administrateur a le pouvoir d'ajouter de nouveaux utilisateurs ou même de supprimer ceux existants, offrant ainsi un contrôle total sur la plateforme.

Le code Javascript de cette page :

```
function ajouterUtilisateur() {
      var username = document.getElementById("newUtilisateurUsername").value;
      var password = document.getElementById("newUtilisateurPassword").value;
      localStorage.setItem(username, password);
      alert("L'utilisateur a ete ajoute avec succes !");
      document.getElementById("newUtilisateurUsername").value = "";
      document.getElementById("newUtilisateurPassword").value = "";
      afficherUtilisateurs();
    }

    function supprimerUtilisateur(username) {
      localStorage.removeItem(username);
      alert("L'utilisateur a ete supprime avec succes !");
      afficherUtilisateurs();
    }
```

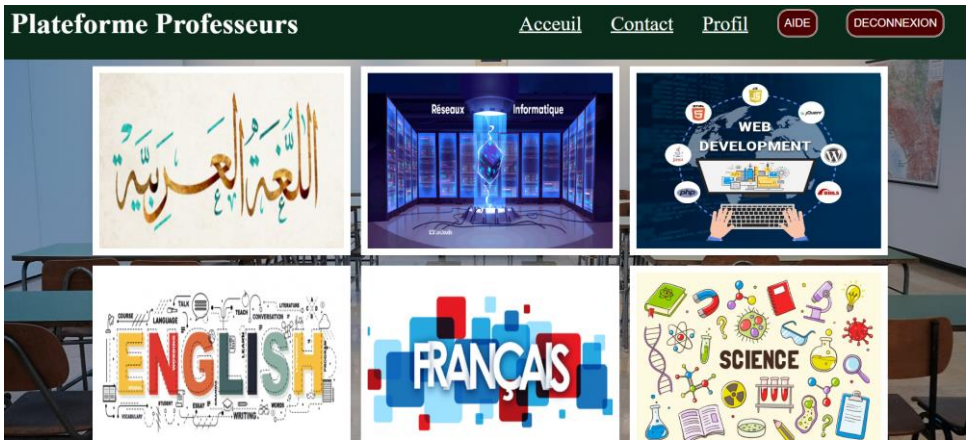
-La fonction ajouterUtilisateur ajoute un nouvel utilisateur au stockage local du navigateur Web. Elle récupère les valeurs du nom d'utilisateur et du mot de passe à partir de deux champs de saisie avec les identifiants "newUtilisateurUsername" et "newUtilisateurPassword", respectivement. Ensuite, elle stocke le nom d'utilisateur et le mot de passe en tant que paire clé-valeur dans le stockage local. Après cela, elle affiche un message d'alerte pour informer l'utilisateur que le nouvel utilisateur a été ajouté avec succès. Enfin, elle efface les champs de saisie et appelle la fonction afficherUtilisateurs.

La fonction supprimerUtilisateur supprime un utilisateur du stockage local du navigateur Web. Elle prend un nom d'utilisateur en argument et supprime la paire clé-valeur correspondante du stockage local. Ensuite, elle affiche un message d'alerte pour informer l'utilisateur que l'utilisateur a été supprimé avec succès. Enfin, elle appelle la fonction afficherUtilisateurs.

```
function afficherUtilisateurs() {
  var table = document.getElementById("utilisateursTable");
  table.innerHTML = "";
  for (var i = 0; i < localStorage.length; i++) {
    var username = localStorage.key(i);
    var password = localStorage.getItem(username);
    var tr = document.createElement("tr");
    var td1 = document.createElement("td");
    td1.innerHTML = username;
    tr.appendChild(td1);
    var td2 = document.createElement("td");
    td2.innerHTML = password;
    tr.appendChild(td2);
    var td3 = document.createElement("td");
    var btnSupprimer = document.createElement("button");
    btnSupprimer.innerHTML = "Supprimer";
    btnSupprimer.onclick = function() {
      supprimerUtilisateur(username);
    };
    td3.appendChild(btnSupprimer);
    tr.appendChild(td3);
    table.appendChild(tr);
  }
}
```

Cette fonction affiche la liste des utilisateurs stockés dans le stockage local du navigateur Web. Elle commence par récupérer une référence à un élément de table avec l'identifiant "utilisateursTable" et efface son contenu. Ensuite, elle parcourt toutes les entrées du stockage local en utilisant une boucle for. Pour chaque entrée, elle récupère le nom d'utilisateur et le mot de passe à partir du stockage local. Puis, elle crée une nouvelle ligne de table (tr) et trois cellules de table (td). La première cellule contient le nom d'utilisateur, la deuxième contient le mot de passe et la troisième contient un bouton "Supprimer". Lorsque ce bouton est cliqué, il appelle la fonction supprimerUtilisateur avec le nom d'utilisateur en argument. Enfin, la fonction ajoute la nouvelle ligne à la table.

❖ ***Page des modules :***



La page en question offre une expérience utilisateur sécurisée et personnalisée. Elle contient 6 photos, chacune représentant un module différent. Les professeurs peuvent accéder à leur propre module en entrant leur code personnel. Cela garantit que seuls les professeurs autorisés peuvent accéder aux modules qui leur sont attribués. De plus, la page offre la possibilité aux utilisateurs de mettre à jour leur nom d'utilisateur et leur mot de passe en cliquant sur l'option "Profil". Cette fonctionnalité permet aux utilisateurs de maintenir la sécurité de leurs informations en mettant régulièrement à jour leurs identifiants de connexion. En somme, cette page offre une expérience utilisateur pratique et sécurisée pour les professeurs.

Le code JAVASCRIPT de cette page :

```
function code1() {
    var code = prompt("Veuillez entrer le code :");;
    if (code == "123") {
        alert("Code valide, redirection vers la page suivante.");
        window.location.href="arabe.html";
        return true;
    } else {
        alert("Code incorrect.");
        return false;
    }
}   function code2() {
    var code = prompt("Veuillez entrer le code :");;
    if (code == "456") {
        alert("Code valide, redirection vers la page suivante.");
        window.location.href="reseau.html";
        return true;
    } else {
        alert("Code incorrect.");
        return false;
    }
}   function code3() {
    var code = prompt("Veuillez entrer le code :");;
    if (code == "789") {
        alert("Code valide, redirection vers la page suivante.");
        window.location.href="admin.html";
        return true;
    } else {
        alert("Code incorrect.");
        return false;
    }
```

```
function code4() {
    var code = prompt("Veuillez entrer le code :");;
    if (code == "101112") {
        alert("Code valide, redirection vers la page suivante.");
        window.location.href="anglais.html";
        return true;
    } else {
        alert("Code incorrect.");
        return false;
    }

}
function code5() {
    var code = prompt("Veuillez entrer le code :");;
    if (code == "131415") {
        alert("Code valide, redirection vers la page suivante.");
        window.location.href="francais.html";
        return true;
    } else {
        alert("Code incorrect.");
        return false;
    }
}
function code6() {
    var code = prompt("Veuillez entrer le code :");;
    if (code == "161718") {
        alert("Code valide, redirection vers la page suivante.");
        window.location.href="science1.html";
        return true;
    } else {
        alert("Code incorrect.");
        return false;
    }
}
```

Ce code contient six fonctions JavaScript: code1, code2, code3, code4, code5 et code6. Chacune de ces fonctions demande à l'utilisateur d'entrer un code en utilisant la fonction prompt. Si le code entré est correct, la fonction affiche un message d'alerte pour informer l'utilisateur que le code est valide et redirige l'utilisateur vers une nouvelle page en utilisant la propriété window.location.href. Si le code entré est incorrect, la fonction affiche un message d'alerte pour informer l'utilisateur que le code est incorrect.

Chaque fonction vérifie un code différent et redirige l'utilisateur vers une page différente si le code est correct. Par exemple, la fonction code1 vérifie si le code entré est "123" et redirige l'utilisateur vers la page "arabe.html" si c'est le cas. La fonction code2 vérifie si le code entré est "456" et redirige l'utilisateur vers la page "reseau.html" si c'est le cas, et ainsi de suite.

❖*Modifier le profil de l'utilisateur :*

Cette page offre à l'utilisateur la possibilité de mettre à jour son profil en entrant d'abord son nom d'utilisateur et son mot de passe actuels corrects. Une fois cette étape accomplie, il peut saisir les nouveaux paramètres, donnant ainsi une touche de fraîcheur à son profil.

Le code JAVASCRIPT de cette page :

```
function modifierUtilisateur() {
  var username = document.getElementById("utilisateurUsername").value;
  var password = document.getElementById("utilisateurPassword").value;
  var newUsername = document.getElementById("newUtilisateurUsername").value;
  var newPassword = document.getElementById("newUtilisateurPassword").value;
  var storedPassword = localStorage.getItem(username);

  if (storedPassword === password) {
    localStorage.setItem(newUsername, newPassword);
    localStorage.removeItem(username);
    alert("Les informations de l'utilisateur ont été modifiées avec succès !");
    document.getElementById("utilisateurUsername").value = "";
    document.getElementById("utilisateurPassword").value = "";
    document.getElementById("newUtilisateurUsername").value = "";
    document.getElementById("newUtilisateurPassword").value = "";
  } else {
    alert("Le nom d'utilisateur ou le mot de passe est incorrect !");
  }
}
```

Voici une explication détaillée du fonctionnement du code :

1-Les variables username, password, newUsername et newPassword sont déclarées et initialisées en récupérant les valeurs des champs de saisie correspondants dans le document HTML.

2-La variable storedPassword est initialisée en récupérant la valeur associée à la clé username à partir du stockage local (localStorage). Cela suppose que le mot de passe de chaque utilisateur est stocké localement sous la forme d'une paire clé-valeur, où la clé est le nom d'utilisateur et la valeur est le mot de passe.

3-Une condition if est utilisée pour vérifier si le mot de passe stocké (storedPassword) correspond au mot de passe saisi par l'utilisateur (password). Si la correspondance est vérifiée, cela signifie que l'utilisateur a fourni les informations d'identification correctes.

4-À l'intérieur de la condition if, la nouvelle paire clé-valeur (newUsername, newPassword) est ajoutée au stockage local à l'aide de la méthode localStorage.setItem(). Ensuite, la paire clé-valeur correspondant à l'ancien nom d'utilisateur (username) est supprimée à l'aide de localStorage.removeItem().

5-Une alerte est affichée pour indiquer à l'utilisateur que les informations ont été modifiées avec succès.

6-Enfin, les champs de saisie sont réinitialisés en vidant leurs valeurs à l'aide de document.getElementById().value = "". Cela permet à l'utilisateur de saisir de nouvelles informations sans avoir à effacer manuellement les champs.

Si la correspondance entre le mot de passe stocké et le mot de passe saisi n'est pas vérifiée, une alerte est affichée pour informer l'utilisateur que le nom d'utilisateur ou le mot de passe est incorrect.

❖ ***Un exemple d'un module (module développement web) :***

Une fois que le professeur se connecte avec son mot de passe, il est dirigé vers un module où il peut accéder à une liste de ses étudiants, affichant leur nom, prénom, numéro de téléphone et statut de paiement. Le professeur a la possibilité d'ajouter de nouveaux étudiants, de supprimer des étudiants existants ou de réinitialiser entièrement la liste en cliquant sur le bouton "Réinitialiser le tableau". Mais ce n'est pas tout ! Le professeur peut également personnaliser son expérience en choisissant parmi trois fonds d'écran : gris (fond d'écran par défaut), bleu ou doré. Il suffit au professeur de sélectionner son fond d'écran préféré et de cliquer sur le bouton "Changer le fond d'écran" pour l'appliquer. Cette fonctionnalité ajoute une touche esthétique à l'interface et permet au professeur de personnaliser son environnement de travail.

Le code JAVASCRIPT de cette page :

```javascript
// Vérifie si les données ont été stockées localement, sinon initialise un tableau vide
var students = JSON.parse(localStorage.getItem("students3")) || [];

// Remplissage initial de la table avec les données stockées
var table = document.getElementById("studentsTable").getElementsByTagName('tbody')[0];
for (var i = 0; i < students.length; i++) {
    var student = students[i];
    var row = table.insertRow();
    row.insertCell().innerHTML = student.firstName;
    row.insertCell().innerHTML = student.lastName;
    row.insertCell().innerHTML = student.telephone;
    row.insertCell().innerHTML = student.paid ? "Oui" : "Non";
    var deleteButtonCell = row.insertCell();
    var deleteButton = document.createElement("button");
    deleteButton.innerHTML = "Supprimer";
    deleteButton.onclick = (function(student) {
        return function() {
            var index = students.indexOf(student);
            students.splice(index, 1);
            localStorage.setItem("students3", JSON.stringify(students));
            location.reload();
        }
    })(student);
    deleteButtonCell.appendChild(deleteButton);
}
```

1-La variable students est initialisée en récupérant les données stockées dans le stockage local (localStorage) sous la clé "students3". Les données sont converties de la forme JSON en un tableau d'objets à l'aide de JSON.parse(). Si aucune donnée n'est présente, un tableau vide est assigné à la variable students à l'aide de l'opérateur || [].

2-La table HTML est récupérée à l'aide de document.getElementById("studentsTable"), en ciblant l'élément qui a l'ID "studentsTable". Ensuite, le corps de la table (tbody) est récupéré en utilisant getElementsByTagName('tbody')[0] et assigné à la variable table.

3-Une boucle for est utilisée pour itérer à travers chaque étudiant dans le tableau students.

4-À chaque itération de la boucle, les informations de l'étudiant courant sont récupérées dans la variable student.

5-Une nouvelle ligne (<tr>) est insérée dans le corps de la table à l'aide de table.insertRow() et assignée à la variable row.

6-Pour chaque propriété de l'étudiant, une nouvelle cellule (<td>) est insérée dans la ligne row. Les propriétés affichées sont firstName, lastName, telephone et paid (statut de paiement de l'étudiant). Les valeurs correspondantes sont insérées dans les cellules en utilisant row.insertCell().innerHTML.

7-Une nouvelle cellule est insérée dans la ligne row pour le bouton de suppression. Un élément de bouton (<button>) est créé en utilisant document.createElement("button") et assigné à la variable deleteButton. Le texte du bouton est défini comme "Supprimer" en utilisant deleteButton.innerHTML = "Supprimer".

8-La fonction de suppression est définie et attribuée à l'événement onclick du bouton. Une fonction anonyme est utilisée pour capturer la valeur actuelle de l'étudiant à supprimer. La fonction retourne une autre fonction qui effectue la suppression lorsqu'elle est appelée. Elle trouve l'index de l'étudiant dans le tableau students, le supprime à l'aide de students.splice(index, 1) et met à jour les données dans le stockage local en utilisant localStorage.setItem("students3", JSON.stringify(students)). Enfin, location.reload() est appelée pour recharger la page et afficher les changements.

9-Le bouton de suppression est ajouté à la cellule de suppression en utilisant deleteButtonCell.appendChild(deleteButton).

Ce code permet de remplir dynamiquement la table avec les données des étudiants et d'ajouter des boutons de suppression pour chaque étudiant. Lorsqu'un bouton de suppression est cliqué, l'étudiant correspondant est supprimé du tableau et les modifications sont sauvegardées dans le stockage local.

```
function addStudent() {
    // Récupération des valeurs des champs de saisie
    var firstName = document.getElementById("firstName").value;
    var lastName = document.getElementById("lastName").value;
    var telephone = document.getElementById("telephone").value;
    var paid = document.getElementById("paid").checked;

    // Création d'un nouvel étudiant et ajout dans le tableau et dans le tableau local
    var student = { firstName: firstName, lastName: lastName, telephone: telephone, paid: paid };
    students.push(student);
    localStorage.setItem("students3", JSON.stringify(students));
    var row = table.insertRow();
    row.insertCell().innerHTML = firstName;
    row.insertCell().innerHTML = lastName;
    row.insertCell().innerHTML = telephone;
    row.insertCell().innerHTML = paid ? "Oui" : "Non";
    var deleteButtonCell = row.insertCell();
    var deleteButton = document.createElement("button");
    deleteButton.innerHTML = "Supprimer";
    deleteButton.onclick = (function(student) {
        return function() {
            var index = students.indexOf(student);
            students.splice(index, 1);
            localStorage.setItem("students3", JSON.stringify(students));
            location.reload();
        }
    })(student);
    deleteButtonCell.appendChild(deleteButton);
}
```

1-Les valeurs des champs de saisie (firstName, lastName, telephone et paid) sont récupérées en utilisant document.getElementById().value. Ces valeurs correspondent aux informations saisies par l'utilisateur pour le nouvel étudiant.

2-Un nouvel objet student est créé en utilisant les valeurs récupérées des champs de saisie. Cet objet représente le nouvel étudiant avec les propriétés firstName, lastName, telephone et paid.

3-L'étudiant nouvellement créé est ajouté à la fin du tableau students en utilisant students.push(student).

4-Les données mises à jour dans le tableau students sont enregistrées dans le stockage local en utilisant localStorage.setItem("students3", JSON.stringify(students)). Les données sont converties en format JSON à l'aide de JSON.stringify() avant d'être stockées.

5-Une nouvelle ligne (<tr>) est insérée dans la table existante (table) pour afficher les informations du nouvel étudiant. Des cellules (<td>) sont insérées dans cette ligne pour chaque propriété de l'étudiant en utilisant row.insertCell().innerHTML.

6-Une nouvelle cellule est insérée dans la ligne pour le bouton de suppression. Un élément de bouton (<button>) est créé et configuré de la même manière que dans le code précédent pour la suppression. La fonction de suppression est attachée à l'événement onclick du bouton.

7-Le bouton de suppression est ajouté à la cellule de suppression.

Cette fonction permet donc d'ajouter un nouvel étudiant à la fois à la fois dans le tableau affiché dans la page HTML et dans le stockage local. En cliquant sur le bouton de suppression d'un étudiant nouvellement ajouté, l'étudiant est supprimé du tableau et du stockage local, et la page est rechargée pour refléter les modifications.

```
function resetTable() {
    // Vide le tableau et le tableau local
    table.innerHTML = "";
    students = [];
    localStorage.setItem("students3", JSON.stringify(students));
}
```

1-La méthode innerHTML est utilisée pour vider le contenu de la table. En affectant une chaîne de caractères vide à table.innerHTML, toutes les lignes et cellules existantes dans la table sont supprimées.

2-La variable students est réinitialisée en assignant un tableau vide à cette variable. Cela permet de vider complètement le tableau des étudiants.

3-Les données mises à jour, avec le tableau vidé, sont enregistrées dans le stockage local en utilisant localStorage.setItem("students3", JSON.stringify(students)). Ici, un tableau vide est converti en format JSON à l'aide de JSON.stringify() et stocké sous la clé "students3".

Ainsi, lors de l'appel de cette fonction, la table des étudiants sera réinitialisée en supprimant toutes les données affichées et en vidant le tableau stocké localement.

```
function changeBackground() {
    // Récupération de la valeur du selecteur de fond d'écran
    var background = document.getElementById("background").value;

    // Changement du fond d'écran
    document.body.style.backgroundImage = "url(" + background + ")";
    localStorage.setItem("background3", background);
}

// Changement du fond d'écran si une valeur est stockée localement
var background = localStorage.getItem("background3");
if (background) {
    document.body.style.backgroundImage = "url(" + background + ")";
}
```

1-La valeur sélectionnée dans le sélecteur de fond d'écran est récupérée en utilisant document.getElementById("background").value et est stockée dans la variable background.

2-Le fond d'écran de la page est modifié en utilisant document.body.style.backgroundImage. La nouvelle valeur de background est utilisée pour définir l'URL du fond d'écran.

3-La nouvelle valeur de background est également enregistrée dans le stockage local en utilisant localStorage.setItem("background3", background). Cela permet de conserver le fond d'écran sélectionné même après le rechargement de la page.

Ensuite, le code vérifie s'il existe une valeur de fond d'écran stockée localement.

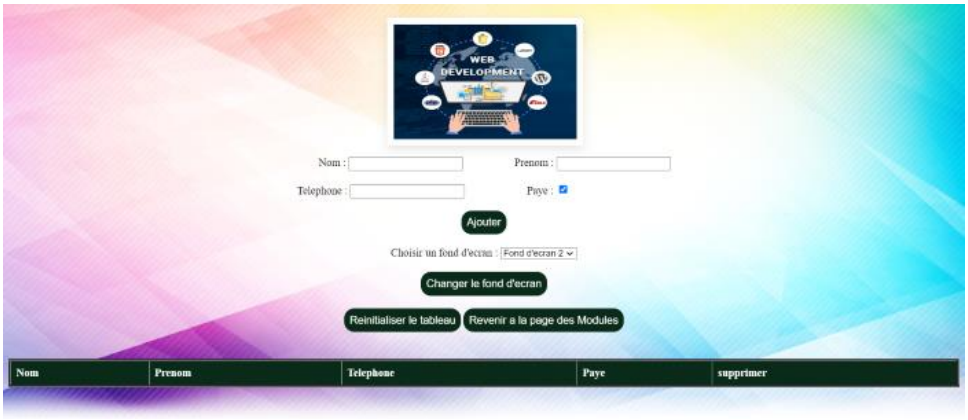
4-La valeur de fond d'écran est récupérée à l'aide de localStorage.getItem("background3") et est stockée dans la variable background.

5-Si une valeur de fond d'écran est présente, le fond d'écran de la page est modifié en utilisant la même logique que dans la fonction changeBackground(). Cela permet de restaurer le fond d'écran précédemment sélectionné lors du chargement de la page.

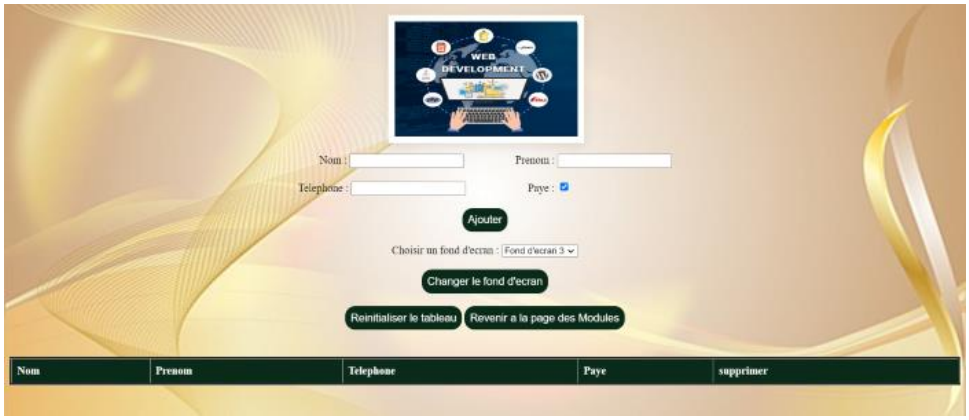
En utilisant cette fonction et le code associé, il est possible de changer et de conserver le fond d'écran de la page en fonction des préférences de l'utilisateur, en utilisant le sélecteur de fond d'écran.

❖ ***Fond écran 1 (gris par défaut) :***

❖ ***Fond écran 2 (bleu) :***



❖ ***Fond écran 3 (doré) :***



❖ ***Maintenance :***

Pour sauvegarder et récupérer les données, j'ai utilisé le stockage local JSON et j'ai manipulé le DOM. Cette approche astucieuse me permet de stocker les informations de manière persistante et d'interagir avec les éléments de la page de manière dynamique.

En utilisant le stockage local JSON, je peux facilement convertir les données en format JSON et les enregistrer sous une clé spécifique. Cela me permet de stocker les informations des étudiants, tels que leur nom, prénom, numéro de téléphone et statut de paiement, de manière organisée et accessible.

Ensuite, grâce à la manipulation du DOM, je peux créer, modifier et supprimer des éléments HTML en fonction des actions de l'utilisateur. Par exemple, lorsque l'utilisateur ajoute un nouvel étudiant, j'ajoute dynamiquement une nouvelle ligne dans la table affichée à l'écran en utilisant insertRow() et insertCell(). De plus, je peux attacher des événements aux boutons de suppression afin de réagir lorsque l'utilisateur souhaite supprimer un étudiant spécifique. Ces interactions dynamiques améliorent l'expérience utilisateur en offrant une interface réactive et en temps réel.

En utilisant cette combinaison de stockage local JSON et de manipulation du DOM, je parviens à créer une application attrayante et fonctionnelle pour la gestion des étudiants. Les données sont sauvegardées de manière fiable, tandis que l'interface utilisateur est interactive et conviviale.

❖*Conclusion :*

❖ *Objectifs atteint :*

Après la réalisation de notre mini-projet, j'ai pu atteindre mes objectifs de manière satisfaisante. Notre plateforme a démontré son efficacité et sa capacité à simplifier la gestion des étudiants. Voici les résultats obtenus :

1-Gain de temps : Grâce à notre outil innovant, j'ai pu automatiser de nombreuses tâches administratives. Les opérations fastidieuses telles que la saisie manuelle des informations des étudiants ont été réduites, ce qui m'a permis de gagner un temps précieux. Je peux désormais enregistrer et suivre les informations des étudiants en quelques clics seulement.

2-Organisation améliorée : Notre plateforme intuitive m'a permis de maintenir une organisation optimale. Les données des étudiants sont stockées de manière structurée et accessible. Je peux facilement consulter les informations de chaque étudiant, telles que leur nom, prénom, numéro de téléphone et statut de paiement, en un seul endroit. Cela a considérablement simplifié la gestion et la recherche d'informations.

3-Réduction des erreurs : Grâce à la saisie automatisée des données et à la validation des champs, j'ai pu minimiser les erreurs humaines. Notre outil vérifie les informations saisies pour s'assurer de leur exactitude, ce qui contribue à maintenir des données précises et fiables.

4-Convivialité et facilité d'utilisation : L'interface intuitive de notre plateforme rend son utilisation agréable et accessible. Même sans connaissances techniques approfondies, j'ai pu facilement naviguer et interagir avec les fonctionnalités offertes. La convivialité de l'outil a grandement facilité son adoption et son intégration dans ma routine quotidienne.

En somme, grâce à notre solution, j'ai pu optimiser la gestion des étudiants et simplifier mes tâches administratives. Les avantages obtenus en termes de gain de temps, d'organisation, de précision des données et de convivialité m'ont permis d'atteindre mes objectifs avec succès. Je suis ravi d'avoir pu mettre en place cette solution innovante qui améliore considérablement l'efficacité de la gestion des étudiants.

❖ ***Perspectives (objectifs qui restent à faire) :***

-Je suis passionné par le domaine du développement web et j'ai un fort désir de me perfectionner davantage. C'est un domaine incroyablement vaste et captivant qui offre de nombreuses opportunités d'apprentissage et d'évolution. Ma passion pour le développement web ne cesse de croître et j'ai hâte d'approfondir mes connaissances et mes compétences dans ce domaine fascinant.

-Je suis conscient que pour continuer à me développer et exceller, je dois poursuivre mes études et me tenir au courant des dernières tendances et technologies du web. Je suis prêt à investir mon temps et mon énergie pour acquérir de nouvelles compétences, explorer de nouveaux langages de programmation, approfondir mes connaissances en conception web et développer une expertise dans des domaines spécifiques tels que le développement front-end, le développement back-end ou le développement d'applications web.

-Le domaine du développement web est en constante évolution, ce qui le rend encore plus passionnant. Chaque jour, de nouvelles idées émergent, de nouvelles techniques sont développées et de nouveaux défis se présentent. Cela signifie que je ne cesserai jamais d'apprendre et que je pourrai toujours me surpasser.

-Je suis déterminé à devenir un développeur web accompli, à créer des expériences en ligne exceptionnelles et à contribuer à l'évolution de cet univers dynamique. Je suis prêt à relever les défis, à me plonger dans de nouveaux projets et à continuer à apprendre et à grandir en tant que professionnel du développement web.

-Le développement web est bien plus qu'un simple métier pour moi, c'est une véritable passion. Je suis motivé, enthousiaste et prêt à m'investir pleinement pour devenir un expert dans ce domaine en constante évolution.